PROJETO E CONCEPÇÃO DE UMA PLATAFORMA PARA ROBÓTICA COGNITIVA EMBARCADA

MARCIO FIGUEIREDO GARCIA*, DIEGO STÉFANO F. FERREIRA*, AUGUSTO LOUREIRO DA COSTA*

**Laboratório de Robótica
Escola Politécnica da Universidade Federal da Bahia
Salvador, Bahia, Brasil*

Emails: marciof14@yahoo.com.br, diego.stefano@gmail.com, augusto.loureiro@ufba.br

Abstract— The Concurrent Autonomous Agent (CAA) is a cognitive architecture composed by three levels that run concurrently: the reactive level, that executes the perception-action cycle using reactive behaviours; the instinctive level, that coordinates plan execution; and the cognitive level, responsible for planning. In this work a hardware architecture dedicated to the execution of the CAA in the omnidirectional robot AxéBot is presented. The work corrects mistakes from previous works and conceives a complete embedded platform for cognitive robotics. The results shows the hardware development and presents an experiment that validates platform performance.

Keywords— Embedded Systems, Robotics, Artificial Intelligence.

Resumo— O Agente Autônomo Concorrente (AAC) é uma arquitetura cognitiva composta por três níveis que executam concorrentemente: o nível reativo, que executa o ciclo percepção-ação do agente utilizando comportamentos reativos; o nível instintivo, que coordena a execução de um plano; e o nível cognitivo, responsável pelo planejamento. Neste trabalho, apresenta-se uma arquitetura de hardware dedicada à execução do AAC no robô omnidirecional AxéBot. O trabalho corrige erros de trabalhos anteriores e concebe uma plataforma embarcada completa para robótica cognitiva. Os resultados mostram o desenvolvimento do hardware e apresentam um experimento que valida o funcionamento da plataforma.

Palavras-chave— Sistemas Embarcados, Robótica, Inteligência Artificial.

1 Introdução

A robótica é uma disciplina interdisciplinar que estuda o desenvolvimento de soluções para problemas em que a ação humana deve ser substituída ou otimizada para acelerar a realização de um processo físico de maneira segura. A natureza dessas soluções permite subdividir a robótica em subáreas não disjuntas: robótica de manipuladores, móvel, teleoperada, autônoma, etc. O presente trabalho tem como tema a robótica móvel autônoma cognitiva, pois desenvolve-se um robô móvel omnidirecional e uma plataforma de hardware para execução de um agente autônomo que governará a atuação autônoma do robô no mundo.

A arquitetura cognitiva utilizada neste trabalho é a do Agente Autônomo Concorrente (AAC), que possui três níveis, o reativo (percepção-ação), o instintivo (coordenação de comportamentos reativos) e o cognitivo (planejamento). Diversos trabalhos já foram desenvolvidos utilizando esta arquitetura, mas o de maior interesse para o presente desenvolvimento é o de (Ferreira et al., 2014).

Em (Ferreira et al., 2014), uma rede de microcontroladores é projetada para embarcar o agente autônomo concorrente no AxéBot. A rede foi projetada para acomodar o modelo cognitivo do agente, consistindo de três microcontroladores: o PSoc para o nível reativo, o mbed LPC1768 para o nível instintivo e o DNP 2486 para o nível cognitivo. A comunicação entre o reativo e o instintivo era feita através de uma rede CAN e a comunicação entre o instintivo e o cognitivo, por sua vez,

usando uma rede Ethernet. A implementação do AAC empregada para os níveis instintivo e cognitivo foi uma adaptação do código da biblioteca Expert-Coop++, desenvolvida inicialmente para ser aplicada em competições de futebol de robôs na categoria de robôs simulados.

Mas a arquitetura desenvolvida apresentou algumas limitações. Segundo o autor (Ferreira et al., 2014), o mbed LPC1768, ao executar o código do Expert-Coop++ para o nível instintivo, apresentou limitações de memória RAM, e o código teve que ser severamente simplificado para executar o motor de inferência deste nível. Assim, o tamanho máximo das bases de conhecimento foi reduzido. A implementação foi baseada em testes de bancada, com o circuito da camada física da rede CAN sem a utilização de transceivers, o que limita a velocidade desta rede, e a rede Ethernet necessitando de um switch externo para conectar o DNP 2486 e o mbed LPC1768.

Este trabalho tem como objetivo conceber uma plataforma robótica completa que não só supra as deficiências da arquitetura anterior, mas também estenda-a. Para lograr a concepção da supramencionada plataforma, que será denominada AxéBot 2.0, uma nova estrutura física para o robô foi construída. Com isso, se fez necessário repetir as tarefas de modelagem e controle para a nova estrutura física. No que tange à arquitetura de hardware, priorizou-se a homogeneidade, substituindo a placa DNP 2486, por uma mbed Renesas GR-PEACH, que também é baseada na arquitetura ARM. Adicionalmente, os microcon-

troladores agora se conectam utilizando um único barramento CAN, com os devidos transceivers instalados para descartar limitações de velocidade,

O restante do trabalho está organizado conforme descrito a seguir. A Seção 2 descreve o modelo cognitivo do Agente Autônomo Aconcorrente. Na Seção 3 o protocolo CAN é apresentado. Em seguida, na Seção 4, o AxéBot 2.0 é descrito, seguida da Seção 5, onde os experimentos e resultados com o robô são apresentados. Finalmente, a Seção 6 apresenta algumas considerações finais.

2 O Agente Autônomo Concorrente

O autor (Bittencourt, 1997) propôs um modelo cognitivo denominado Modelo Genérico para Agentes Cognitivos. Este modelo é composto por três camadas ou níveis que desempenham tarefas em diferentes níveis de abstração simbólica: o nível cognitivo, o nível instintivo e o nível reativo. Em (Costa and Bittencourt, 1999), os autores criaram o Agente Autônomo Concorrente, uma arquitetura cognitiva para agentes autônomos baseada no Modelo Genérico para Agentes Cognitivos. A Figura 1 mostra a visão geral da arquitetura.

Figura 1: Arquitetura geral do Agente Autônomo Concorrente.

As próximas subseções descrevem cada nível em maior detalhe.

2.1 Nível Reativo

O nível reativo interage com o ambiente e executa o ciclo percepção-ação do agente. Nele são implementados comportamentos reativos simples: as direções norte (N), nordeste (NE), leste (L), sudeste (SE), sul (S), sudoeste (SO) e oeste (O) que o robô pode seguir com velocidade constante. Estes comportamentos são implementados por um controlador cinemático de velocidade.

O nível instintivo recebe as leituras de sensores realizadas pelo nível reativo e comunica que comportamento deve estar ativo.

2.2 Nível Instintivo

O nível instintivo possui um Sistema Baseado em Conhecimento (SBC) que com base na meta local enviada pelo nível cognitivo, nas informações sensoriais enviadas pelo reativo e na sua base de regras infere qual o comportamento deve estar ativo no nível reativo.

A sua base de conhecimento possui uma coleção de bases de regras que são chamadas de *planos*, e a base de regras atual é selecionada pelo nível cognitivo e o seu procedimento de planejamento.

2.3 Nível Cognitivo

O nível cognitivo é onde o planejamento acontece. Este nível também possui um SBC, mas dessa vez o mesmo é utilizado como mecanismo de busca para o algoritmo de planejamento. Um modelo lógico do mundo é mantido e sua base de conhecimento divide-se em local e social, pois este nível também é responsável por estabelecer comunicação com outros agentes.

3 O Protocolo CAN

O hardware do sistema robótico concebido neste trabalho utiliza barramentos CAN (*Controller Area Network*) para estabelecer a comunicação entre os seus nós computacionais. A especificação CAN, assim como outras especificações de protocolos de comunicação digital, pode ser descrita conforme o modelo de sete camadas OSI (*Open System Interconnection*) que, por sua vez, é aplicado para modelar a comunicação entre aplicações executando em dispositivos conectados em rede (Briscoe, 2000).

O protocolo CAN especifica apenas a camada de enlace e parte da camada física do modelo OSI (Figura 2). Na camada de enlace são definidas duas subcamadas, uma delas sendo a LLC (*Logical Link Control*, ou Controle Lógico do Enlace) responsável pela filtragem na recepção de quadros, notificação de sobrecarga e gestão da recuperação. A outra subcamada da camada de enlace, a MAC (*Medium Access Control*, ou Controle de Acesso ao Meio), assume as tarefas de encapsulamento e desencapsulamento dos dados, codificação dos quadros, detecção e sinalização de erros, e serialização/desserialização. A parte da camada física contemplada pela especificação CAN corresponde à criação do sinal (codificação e decodificação dos sinais elétricos em bits, tempos de bits e sincronização), que estabelece os requisitos que o meio físico de transporte e os *transceivers* devem atender (Richards, 2002).

Figura 2: O protocolo CAN e o modelo OSI (Corrigan, 2002).

O restante da especificação da camada física, isto é, a Conexão com o Meio Físico (em inglês, *Physical Medium Attachment*, ou PMA) e a Interface Dependente do Meio (*Medium Dependent Interface* ou MDI), são definidos no padrão ISO 11898, que também engloba a especificação CAN descrita acima (Richards, 2002). O sinal elétrico diferencial da rede CAN é mostrado na Figura 3.

Figura 3: Sinal diferencial do barramento CAN (Richards, 2002).

A arbitragem do barramento é responsabilidade do controlador CAN de cada nó e, por conseguinte, é transparente à camada de aplicação. O nível lógico baixo ('0') é o dominante do barramento, e todos os nós leem o barramento enquanto enviam mensagens. Utilizando estas duas premissas a resolução de colisão procede da seguinte maneira:

- Sejam dois nós A e B que percebem o barramento livre e tentam transmitir ao mesmo tempo;
- Em um dado momento, o nó A tentar colocar o barramento em nível recessivo ('1') enquanto o nó B tentar colocar o barramento em nível dominante ('0');

- O nível dominante prevalece no barramento, e o nó A sente que o barramento está em '0' enquanto ele enviou '1', percebe que perdeu na arbitragem do barramento e aborta o seu envio.

Assim, mensagens com mais zeros (ou seja, mais bits dominantes) no início, possuem maior prioridade (Corrigan, 2002).

4 Arquitetura Proposta

Uma visão geral da arquitetura proposta pode ser vista na Figura 4. Nesta figura nota-se a presença de dois barramentos CAN: um entre os níveis reativo (PSoC 5LP) e instintivo (mbed LPC1768), e outro entre o instintivo e o cognitivo (Renesas GR-PEACH). Isso se faz necessário porque o primeiro barramento precisa comportar o tráfego de informações o ciclo percepção-ação, que deve ser perpétuo. Já o segundo corresponde à coordenação da execução do planejamento. Se apenas um barramento fosse utilizado, as colisões tornariam o comportamento do sistema imprevisível.

Uma placa de circuito impresso comportará todos os elementos do sistema e, portanto, toda a comunicação será embarcada: a camada física dos barramentos consistirá de transceivers CAN e as linhas de transmissão do barramento (tal como ilustrado na Figura 2), estas últimas implementadas por trilhas da placa.

As subseções seguintes descreverão detalhes de implementação de cada nível do agente em cada nó computacional da arquitetura.

4.1 Nível Reativo: PSOC 5LP

O nível reativo foi embarcado no PSOC (*Programmable System-on-a-Chip*) 5LP, da *Cypress Semiconductor*. O PSOC 5LP é adequado ao nível reativo tanto por ser reconfigurável em nível de *hardware* quanto por oferecer múltiplas opções de periféricos de comunicação, aquisição de dados, sistemas digitais, etc. (Semiconductor, 2015).

O PSOC possui um processador ARM Cortex-M3 de 32-bits com 67MHz, memória flash de 256KB, memória RAM de 64KB e EEPROM de 2KB. Além disso, o PSOC implementa oferece diversas opções de protocolos de comunicação (I2C, SPI, UART entre outros), incluindo aquele de maior relevância para o presente trabalho: o protocolo CAN 2.0b (Semiconductor, 2015).

Sendo o nível reativo responsável pelo ciclo percepção-ação do agente, no PSOC também foi embarcado o controlador cinemático do AxéBot. O controlador descrito por Ferreira et al. (2014) foi utilizado, com parâmetros dimensionais adaptados para a versão atual do AxéBot.

Como se vê na Figura 4, um barramento CAN é empregado para a comunicação entre este nível e o instintivo, implementado em um ARM mbed

Figura 4: Arquitetura de hardware proposta.

LPC1768. O barramento foi configurado com velocidade máxima de transmissão, isto é, 1Mbps, e os recursos de filtragem de mensagens pelo campo ID dos *frames* foram aplicados para identificar a informação a ser transmitida e, ao mesmo tempo, atribuir prioridades às mensagens.

4.2 Nível Instintivo: ARM mbed LPC1768

O nível instintivo foi embarcado em um ARM mbed LPC1768. Este módulo possui um microcontrolador NXP LPC1768 que, por sua vez, contém um microprocessador ARM Cortex-M3.

Neste nível um SBC tem sua base de fatos alterada tanto por informações sobre o ambiente vindas do nível reativo quanto por informações relativas ao planejamento vindas do nível cognitivo, sendo essas informações trocadas através de barramentos CAN distintos. No barramento CAN instintivo/cognitivo, o instintivo recebe metas locais e envia informações simbólicas sobre o estado do ambiente; no barramento instintivo/reativo, o reativo envia dados crus sobre o estado do agente e do ambiente e o instintivo processa esses dados, atualiza a base de conhecimento do seu SBC e envia o comportamento selecionado pelo motor de inferência deste último ao reativo.

Neste nível também é utilizado o recurso de filtragem do barramento CAN para estabelecer a natureza das mensagens e as suas prioridades.

4.3 Nível Cognitivo: ARM mbed Renesas GR-PEACH

O módulo ARM mbed Renesas GR-PEACH é o de maior desempenho computacional dos três: possui um processador ARM Cortex-A9 de 400MHz, 10MB de memória RAM e 8MB de memória flash.

O alto poder computacional desse módulo faz-se um requisito para executar o nível cognitivo por conta da tarefa que o mesmo executada: planejamento.

Neste trabalho, um planejador clássico baseado em um SBC foi implementado para esse nível. O algoritmo utilizado é alimentado com uma base de conhecimento escrita na linguagem de domínio específico do AAC e realiza uma busca no espaço

de estados para encontrar o plano, que é transmitido para o nível instintivo na forma de metas locais através da rede CAN que interconecta os dois níveis.

5 Resultados

Os resultados obtidos são divididos em duas partes: a concepção do hardware do AxéBot 2.0 e a validação da arquitetura através de experimentos utilizando o software do AAC atuando na supracitada arquitetura de hardware.

5.1 Concepção da Placa de Circuito Impresso

Inicialmente as dimensões e formato da placa de circuito impresso foram desenvolvidas no software Autodesk Autocad 2D e depois importadas para o ambiente de desenvolvimento do software Proteus ARES, definindo assim a borda da placa. A presente plataforma tem formato circular e com três "cavas" onde ficam os espaços das três rodas omnidirecionais do robô AxéBot. Na Figura 5 vê-se essas dimensões.

Figura 5: Layout da placa de circuito impresso.

O Proteus ARES Design Suite foi o software usado na montagem da placa e projeto da PCB. O Recurso utilizado para o roteamento das trilhas foi o autorouter, roteamento baseado em formato padrão. O roteador do ARES usa algoritmo

mos avançados de redução de conflitos baseados em custo, comprovados para maximizar as taxas de conclusão mesmo nas placas mais densamente compactadas, ou seja, com placas com mais camadas internas. Após a configuração e execução desse procedimento para o roteamento da placa de 4 camadas (Top Layer, Inner1, Inner2 e Botton Layer) um ajuste manual foi necessário para retirar trilhas desnecessárias. O ajuste automático dos componentes não foi realizado, preferindo assim a localização das plataformas de forma manual a fim de obter melhor posição devido ao tamanho reduzido da placa, por questões do grande número de componentes. Também foi utilizado o sistema de regras de design, que impõe o roteamento de trilhas compatíveis e a verificação e correção de violações dos pontos críticos antes da geração dos arquivos Gerber, no formato RS274X.

O desenvolvimento levou em consideração padrões internacionais importantes adotados com relação a distância (*clearance*) da malha de terra para as trilhas de sinal, devido a PCB ter um circuito em que a tensão pode exceder os 200V, por utilizar um circuito de carregamento de capacitores e por circular uma corrente próximo dos 40A no momento de acionamento do circuito de disparo através de uma solenoide. Sob a influência dessa tensão e sobre o grau de sujeira da PCB a distância necessária cresce com a tensão e o grau de poluição/umidade.

5.2 Agente Autônomo Concorrente Embarcado

Para a validação da execução do AAC na placa descrita na subseção anterior (mostrada na Figura 7) é suficiente a realização de uma tarefa simples de planejamento que permita avaliar o funcionamento dos SBC's nos níveis instintivo e cognitivo, do controlador no nível reativo e da comunicação entre os níveis.

Assim, para o planejamento, utilizou-se a seguinte base de conhecimento simples:

```
(ir_para_p2
  (if (logic(robo posicao p1)))
  (then (logic(robo posicao p2))))
```

Esta base de conhecimento, escrita na linguagem de domínio específico do AAC, gera um plano contendo apenas uma ação: ir do ponto *p1* para o ponto *p2*.

p1 e *p2* são dadas por $[0m, 0m]$ e $[10m, 10m]$, e o robô está inicialmente em *p1*. A base de conhecimento do nível instintivo é aquela utilizada por (Ferreira et al., 2014), que considera o mundo dividido em zonas (norte, nordeste, leste, sudeste, sul, sudoeste, oeste e noroeste) em torno do robô.

Um exemplo de regra nesse mundo é dado por:

```
(r11
  (if (logic(robo zona segura))
    (logic(obstaculo zona nordeste))
    (logic(meta zona nordeste)))
  (then (logic(robo acao norte))))
```

Na regra *r11* acima a zona segura significa que não há nenhum obstáculo a menos de 20cm do robô. A regra testa ainda se a base de fatos contém os fatos (*logic(obstaculo zona nordeste)*) (o obstáculo mais próximo está a nordeste) e (*logic(meta zona nordeste)*) (a meta está na zona nordeste), selecionando o comportamento *norte* no reativo caso esses fatos existam.

O testes foram realizados com os motores girando a vazio. Os resultados da navegação são mostrados nas Figuras 8 e 9.

6 Considerações Finais

Neste trabalho foi apresentada uma arquitetura de hardware completa para o robô móvel omnidirecional AxéBot destinada à execução do modelo cognitivo do Agente Autônomo Concorrente. O presente projeto vai além de tentativas anteriores (Ferreira et al., 2014), concebendo uma placa de circuito impresso com as dimensões do robô e, conforme os experimentos, apta à execução do AAC.

O hardware descrito aqui, porém, consiste apenas de um subconjunto das suas capacidades: aquelas necessárias à execução do AAC. Outros elementos não descritos, como a possibilidade de comunicação com outros agentes embarcados através de um módulo WiFi ou de utilizar outras plataformas computacionais, mostram que essa plataforma é mais genérica e pode ser aplicada como base para o estudo de tópicos mais avançados, como sistemas multi-agentes.

Como trabalhos futuros, pretende-se utilizar a plataforma em um robô agindo autonomamente em um ambiente dinâmico.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Referências

- Bittencourt, G. (1997). In the quest of the missing link, *International Joint Conference of Artificial Intelligence*.
- Briscoe, N. (2000). Understanding the osi 7-layer model, *PC Network Advisor* **120**(2).

Figura 6: As quatro camadas da placa.

Figura 7: Placa confeccionada.

Figura 8: Resultados da navegação na componente x .

Corrigan, S. (2002). Introduction to the controller area network (can), *Application Report*

Figura 9: Resultados da navegação na componente y .

SLOA101 pp. 1–17.

Costa, A. L. d. and Bittencourt, G. (1999). From a concurrent architecture to a concurrent autonomous agents architecture, *Lecture Notes in Artificial Intelligence* **1856**: 85–90.

Ferreira, D. S. F., Silva, R. R. d. and Costa, A. L. d. (2014). Embedding a concurrent autonomous agent in a microcontrollers network, *Biosignals and Biorobotics Conference (2014): Biosignals and Robotics for Better and Safer Living (BRC), 5th ISSNIP-IEEE*, IEEE, pp. 1–6.

Richards, P. (2002). A can physical layer discussion, *Microchip Technology Inc.*

Semiconductor, C. (2015). Psoc® 5lp: Cy8c58lp family datasheet, *Prod. Cypress Semiconductor*.